UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Detectar y prevenir actividades y contenido ilegal en internet

Jesús Rubén Gastón González

Tutor: Gustavo Sutter Capristo

Julio 2016

Resumen

Este Trabajo de Fin de Grado se centra en el desarrollo de un sistema capaz de detectar y prevenir actividades ilegales a partir de la información publicada en las redes sociales.

El nombre del proyecto es SAMi2. Este nombre proviene de **Semantics Analysis Monitor** for the **Illegal** Use of **Internet**. Es un proyecto creado por la empresa HI Iberia y financiado por el Directorado General Europeo para Asuntos Internos en el marco del Programa para la Prevención de/y Lucha contra el Crimen.

SAMi2 recopila las publicaciones que envían los usuarios a través de la red social Twitter con el fin de analizarlas lo más rápido posible para poder detectar una posible situación de peligro.

Analizando estos datos a los pocos minutos de que sean publicados, se puede obtener información muy relevante para los cuerpos de seguridad, ya que ellos sabrán de antemano a la situación que se exponen pudiendo actuar en consecuencia de la manera más eficaz para resolver una posible situación conflictiva o en el mejor de los casos, evitarla.

El objetivo principal es ayudar a los agentes encargados de buscar en las redes sociales este tipo de contenido sugiriendo tuits con palabras clave que anteriormente se establecieron en un perfil de búsqueda. De esta manera se puede abarcar un mayor contenido publicado por los usuarios de Twitter y únicamente el agente encargado de esta búsqueda estará realizando una búsqueda más en profundidad, obteniendo resultados más precisos ya que este sistema ha realizado un primer filtrado.

Palabras clave

Trabajo de final de grado, SAMi2, análisis semántico del uso ilegal de internet, analizar tuits, Twitter, tuit, fuerzas de seguridad, detectar peligro.

Abstract

This end-of-degree proyect focuses on the development of a system that can detect and prevent illegal activities from information published on social networks.

The project name is SAMi2. This name comes from **Semantics Analysis Monitor** for the **Illegal** Use of **Internet**. It is a project created by the company HI Iberia and funded by the European General Directorate for Internal Affairs under the Program for the Prevention of/and Crime Control.

SAMi2 collects the publications sent by users on the social network Twitter in order to analyze them as quickly as possible to detect a potentially hazardous situation.

Analyzing these data within a few minutes that are published, you can get obtain very important information to the security forces, because they will know n advance the situation exposed and can act accordingly in the most effective way to resolve a possible situation conflicting or in the best case, avoid it.

The main goal is to help officials responsible for search on social networks such content suggesting tweets with keywords that were previously established. So you can cover major content posted by users of Twitter and only agent in charge of this search will be performing a more in-depth search, obtaining more accurate results because our system has made a first filter.

Keywords

End-of-Degree proyect, SAMi2, Semantics Analysis Monitor for the Illegal Use of Internet, analyze tweets, Twitter, tweet, police, detect hazards.

Agradecimientos

A mi familia y amigos que siempre me han apoyado y gracias a ellos he llegado hasta aquí.

A mis profesores. En especial a los que tienen vocación y han sabido enseñarme, no sólo en el ámbito académico.

A mis compañeros de HI Iberia, por darme la oportunidad de presentar este proyecto realizado allí y por ayudarme a crecer profesionalmente.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

1 Introducción

A principios del curso 2015/2016 la Escuela Politécnica Superior de la Universidad Autónoma de Madrid a través de la asignatura de prácticas en empresa me proporcionó la posibilidad de realizar unas prácticas en una empresa para acercarme al mundo laboral y poner en práctica todos los conocimientos que he ido adquiriendo a lo largo de estos años en la escuela.

La empresa en la que pude realizar las prácticas es HI Iberia [1]. En ella al finalizar el periodo de prácticas curriculares me ofrecieron la posibilidad de continuar con ellos y de este modo podría realizar este Trabajo de Final de Grado presentando el proyecto al que iba a ser asignado.

El proyecto era SAMi2 [2], un proyecto con un usuario final que sería la policía. Lo usarían los agentes que buscan a través de las redes sociales a través de miles y miles de tuits y lo que se quiere lograr es que el sistema sea capaz de entre miles de tuits hacer una selección previa buscando en los textos unas determinadas palabras que se establecerían previamente mediante perfiles. De esta manera los agentes sólo tendrían que buscar entre unos tuits que potencialmente van a ser relevantes. Una vez encontrados las publicaciones que les son más relevantes, podrán indicarlas así en la aplicación para que el sistema “aprenda” y cada vez obtener resultados más precisos.

Cuando me incorporé al proyecto, éste ya estaba iniciado, con las bases de lo que se quería construir y qué tecnologías se querían utilizar, aunque aún no era funcional completamente. Me llamó la atención que un proyecto real se llegase a utilizar unas tecnologías tan modernas, como por ejemplo el uso del framework MeteorJs [3] para desarrollar la parte web. Este framework tiene una corta vida, ya que tuvo su primer lanzamiento en enero de 2012. Esto es algo que no estaba acostumbrados a ver, ya que la mayoría de páginas web que residen y se crean en la red están desarrolladas en otros lenguajes más populares y más longevos como por ejemplo PHP. Otro ejemplo sería la base de datos, en este caso se utilizará una base de datos NoSQL.

1.1 Motivación

Esta memoria de TFG viene motivada para aprovechar la oportunidad de formar parte de un equipo de desarrollo en una empresa, conocer y aprender todo lo posible de un ambiente de trabajo profesional. De esta manera podré irme adaptando para cuando llegue el momento de dar el salto a la vida laboral.

También presentar un proyecto de esta envergadura. Nunca podría imaginar que al tener el primer contacto con una empresa estaría ayudando a desarrollar un proyecto europeo.

Por otra parte, la motivación de desarrollar este proyecto por parte de HI Iberia es poder proporcionar a la policía un recurso más para luchar contra el uso fraudulento de internet. Con este recurso, como se ha mencionado más arriba en la introducción ayudaría a los agentes en su trabajo de buscar en las redes sociales textos ofensivos, inapropiados, que atenten contra las personas… Como resultado se esperaría una disminución de este tipo de contenido en la red e incluso unos índices más bajos de delitos. Otra ventaja de este sistema de detección para desarrollarse es que se podría adaptar a otras lenguas fácilmente para poder utilizarse en países extranjeros si en un futuro se desease.

1.2 Objetivos

El objetivo de este proyecto es crear un sistema que ayude a los agentes de la policía en su labor de búsqueda de contenido ilegal en las publicaciones de la red social Twitter. Esto se hará mediante un primer filtrado de las publicaciones hallando coincidencias con un perfil de búsqueda creado previamente.

El sistema tiene que ser estable, puesto que este sistema será el encargado de recopilar información y analizarla para su posterior visualización, por lo que no se podrá permitir que el sistema se caiga, puesto que se estará perdiendo información que podría resultar de gran ayuda.

Debe ser fácilmente escalable. Como se está analizando una gran cantidad de datos en el menor tiempo posible, se deberá poder contar con un número variable de máquinas dependiendo de la demanda en el procesado de datos.

El sistema debe ser fiable, para que el usuario final lo utilice en su día a día sin temor a un posible fallo o pérdida de datos irrecuperable.

También tiene que ser fácil de usar. El usuario final no tiene por qué tener conocimientos avanzados en informática para poder utilizar este sistema correctamente. Además, este sistema tiene como fin ayudar en la búsqueda en las redes sociales, por lo que no tiene sentido que su uso sea más complicado que la búsqueda directamente en la página web de la red social.

Por último y no menos importante, la interfaz de usuario. Ésta debe ser sencilla y agradable a la vista debido a que los usuarios finales estarán periodos prolongados interactuando con ella.

2 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Estado del arte:** Este capítulo se centra en dar un enfoque detallado del contexto en el que se sitúa el desarrollo de las aplicaciones que forman el sistema a implementar. Se profundizará sobre las tecnologías y herramientas utilizadas.

- **Diseño:** En este siguiente capítulo se abarcará los aspectos relacionados con el diseño del sistema. Se analizará el proceso que sigue un tuit desde el momento en que se obtiene a través de la API de Twitter hasta el momento que queda almacenado en la base de datos con el análisis semántico realizado, disponible para ser mostrado mediante una búsqueda.

  También se mostrará el diseño de la web que será la encargada de mostrar los resultados de una manera y forma que sea intuitiva y agradable para el usuario.

- **Desarrollo:** En este capítulo se encontrarán los aspectos relacionados con el desarrollo e implementación. También los inconvenientes que se han sufrido durante esta etapa tales como errores a la hora de integrar módulos o cambios sufridos en el proyecto ajenos a nuestro control.

- **Pruebas:** En este capítulo se tratará sobre las pruebas realizadas durante todo el proceso de desarrollo. Desde los test unitarios, a test de integración de módulos.

- **Conclusiones:** En este capítulo expondré mis reflexiones después de haber colaborado ampliamente en este proyecto y se propondrán funcionalidades para el futuro e incluso otros posibles usos.

- **Anexos y referencias:** En este último capítulo se tratará sobre los paquetes de software o APIs externos que se han utilizado para que el sistema tenga una mayor funcionalidad y facilidad de uso.

3 Estado del arte

En este segundo capítulo se expondrá un punto de vista más detallado sobre el sistema que se quiere implementar para entender en profundidad la necesidad que representa puesto que no hay ningún sistema similar implementado aún que logre satisfacer esta necesidad.

En esta sección también se tratará sobre las herramientas que se necesitarán para poder desarrollar este sistema eficazmente.

3.1 HI Iberia

Figura 3-1: Logo HI Iberia

HI Iberia es una mediana empresa dedicada a la consultoría tecnológica. Tiene grandes proyectos en marcha, de los cuales la mayoría de ellos son a nivel europeo. Destaca dentro de su sector gracias a su compromiso con la innovación y la capacidad de desarrollar eficazmente nuevos proyectos gracias al personal que posee.

Dado este nivel, tiene el reconocimiento como PYME innovadora por el ministerio.

Figura 3-2: Logo PYME innovadora

Un proyecto de esta magnitud y del cual no se tienen ningún tipo de software precedente o de referencia, solamente documentos técnicos teóricos, presenta un gran reto. En HI Iberia, gracias a sus grandes profesionales, se conseguirá lograr a través de nuevas tecnologías desarrollar este sistema pionero para la policía.

3.2 Usuarios finales

Figura 3-3: logo de la Policía Municipal de Madrid

El usuario final del proyecto será la Policía Municipal de Madrid. Este cuerpo ya tiene agentes que su labor se centra en la búsqueda de textos ofensivos en las redes sociales con el fin de poder evitar actividades ilegales o delitos posteriores. Este grupo de personas se centran en la red social Twitter para realizar sus búsquedas. Actúan como un usuario normal, la diferencia que ellos están trabajando en búsqueda de usuarios que hagan publicaciones que inciten la violencia, humillaciones o generen un peligro inminente.

Para ellos, poder contar con un sistema que haga un filtrado previo sobre una red social que genera cantidades ingentes de contenido es una gran ayuda. Existen páginas web que muestran estadísticas de uso de Twitter [4], que muestran la cantidad de publicaciones en un día en todo el mundo. En el momento en el que escribo estas líneas, cerca de las cinco de la tarde, la web arroja más de 435.000.000 tuits publicados exclusivamente hoy. Esa cantidad nos da una idea de cuanta información se está produciendo alrededor del mundo.

Aunque la finalidad de este sistema no se centra en analizar todas esas publicaciones. Sólo se procesará y se analizarán las que se escriban dentro del territorio español. Pese a esta restricción, se estará tratando con una cantidad de datos muy elevada.

Viendo la cantidad de datos que se generan al cabo de un día, y la tendencia de un uso mayor en estos últimos años, nos podemos percatar de una preocupación bastante extendida en la sociedad sobre qué impacto y que influencia tienen las publicaciones de una determinada persona o grupo. No es extraño oír en las noticias que una persona ha sido detenida por sus comentarios o ciberacoso en las redes sociales. Incluso la propia red social Twitter ya tiene una implementación para limitar el alcance de tuits violentos [5].

En esta época de auge de este tipo de comunicación, que muchas veces se realiza de manera anónima, ya han empezado a surgir publicaciones sobre cómo aprovechar esta información para diferentes fines. Algunos de estos ejemplos son *detectar terremotos con Twitter* [6] *o detectar contenido ofensivo para proteger a los adolescentes* [7]

También se abren discusiones de si se debería moderar o controlar las publicaciones [8].

En este contexto es donde nace SAMi2, una herramienta que empieza a ser necesaria para poder juzgar o "castigar" a los usuarios que hacen un uso ilegal de las redes sociales, un lugar donde gracias al anonimato parecía ser un mundo sin ley.

Con esta herramienta no se pretende amordazar a los usuarios de las redes sociales, sino que una ofensa o acoso que estas personas no realizarían o tuviese una mayor repercusión en la vida real utilicen las redes para cometer estos delitos, pensando que saldrán impunes por el mero hecho de estar en la red.

3.3 Detección de contenido

La detección de contenido en las redes sociales no es un tema muy abarcado, al menos de manera práctica. Siempre que se intenta que un sistema informático simule la capacidad humana, y en este caso la detección de contenido ofensivo o ilegal, se estará hablando de aprendizaje automático.

En la red se puede encontrar muchos documentos teóricos técnicos sobre como abarcar o encaminar el reto a muy alto nivel. En todos ellos se coincide en el uso de clasificadores, como Naive Bayes o redes neuronales.

Un inconveniente añadido, es que para que este sistema sea lo más preciso posible a la hora de realizar las búsquedas y sea capaz de evitar situaciones conflictivas o delitos será necesario que el sistema incluya un análisis semántico de los textos publicados. De esta forma se podrá obtener más información del texto sobre lo que el autor quería transmitir en la publicación. Un ejemplo puede ser si el autor del texto está hablando de una entidad, lugar, persona o en un tiempo determinado.

En este caso se utilizará un paquete de software desarrollado en HI Iberia que se encargará de realizar esta labor. El cometido de este software es ser capaz de realizar análisis sintácticos y semánticos como los que se realizan en las asignaturas de Lengua en los institutos, obteniendo de la frase los diferentes elementos con los que ha sido construida. En la siguiente imagen se podrá ver un sencillo ejemplo de lo que se quiere conseguir.

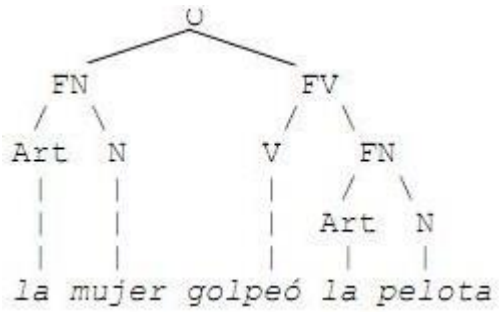

Figura 3-4: Diagrama en árbol de derivación

3.4 Almacenamiento y procesamiento

Para este proyecto se tiene que poder almacenar y procesar una gran cantidad de datos provenientes de Twitter y ser capaz de mostrar esa información en el menor espacio de tiempo posible. Dado este requisito fundamental, se estará delante del fenómeno del Big Data.

Para poder afrontar este reto y obtener el mayor rendimiento a la hora de guardar y mostrar los datos se tendrá que elegir el tipo de base de datos adecuada para este tipo de sistema además del poder de procesamiento necesario para un funcionamiento adecuado. En el capítulo de diseño se discutirá este punto incluyendo datos de rendimiento en las elecciones tomadas.

3.5 Recolección de publicaciones

Para que este sistema que se plantea tenga la funcionalidad apropiada, es fundamental que se nutra de las publicaciones de los usuarios de Twitter. Para este fin, se deberá utilizar la API de Twitter correspondiente para poder tener acceso a los textos que se quieren analizar.

3.6 Protección de datos

Al almacenar contenido publicado en las redes sociales y almacenarlo se tendrá espacial consideración en cumplir con la Ley Orgánica de Protección de Datos. Por lo cual se procederá a anonimizar los datos sensibles recibidos de Twitter para que los textos almacenados en nuestra base de datos no puedan llegar a infringir la ley.

De esta manera, no se podrá saber quien escribió un tuit desde la información de nuestra base de datos, únicamente se obtendrá el texto. Para saber el autor, se deberá acceder a Twitter a través del correspondiente enlace relacionado.

3.7 Tecnologías y herramientas a utilizar

A continuación, se muestran las tecnologías que serán usadas en el proceso de desarrollo del proyecto.

3.7.1 Java

Figura 3-5: Logotipo de java

Java es un lenguaje de programación orientado a objetos especialmente diseñado para tener pocas dependencias de la máquina en la que se está ejecutando. Es un lenguaje altamente portable entre diferentes sistemas y plataformas gracias a que es un lenguaje interpretado por una máquina virtual (JVM).

En este lenguaje se desarrollará la recogida de tuits mediante la API de Twitter, la inserción en la base de datos, la detección de contenido en las publicaciones basada en una red neuronal, el analizador semántico de textos, la funcionalidad para anonimizar datos además de otras funcionalidades y utilidades menos relevantes.

También se incluirá software desarrollado en este mismo lenguaje por la propia empresa HI Iberia para añadir más funcionalidades para mejorar el funcionamiento y usabilidad del sistema final.

3.7.2 Eclipse IDE

Figura 3-6: Logotipo de Eclipse

Eclipse IDE es una herramienta de desarrollo que ayuda a la implementación de una aplicación en Java. Con este entorno de desarrollo se implementará todo el código Java que sea necesario en el proyecto.

3.7.3 MeteorJs

Figura 3-7: Logotipo de Meteor

MeteorJs es un framework escrito en NodeJs (Javascript) para poder construir aplicaciones web y móviles. Su principal ventaja es poder crear aplicaciones rápidamente y poder tener el Front-End (parte del software que interactúa con el cliente) y el Back-end (parte que procesa la entrada del Front-End) en el mismo lenguaje. De esta forma no hay que intercalar varios lenguajes como en otro tipo de diseños.

Un punto a favor es que este framework al estar escrito en Javascript se podrá incluir diversas librerías y código Javascript para mejorar la experiencia de usuario y el aspecto de la web.

Otro aliciente de este framework es que integra MongoDB, esto una base de datos NoSQL que se utilizará en este desarrollo como se verá más adelante.

3.7.4 APIs y librerías Javascript

Figura 3-8: Logotipo de Bootstrap, jquery y API de google maps

Se añadirán diversas librerías y APIs para mejorar el aspecto de la web y la experiencia de usuario. Las principales son:

- Bootstrap: Es un framework Javascript para desarrollar rápidamente y de manera responsive el diseño y el aspecto de una web.

- Jquery: Es una librería Javascript que permite el manejo de eventos, interacción con los documentos HTML, manipular los elementos del DOM y el uso de AJAX entre otros.

- Google Maps APIs: Es una API en Javascript para poder integrar e interactuar con un mapa en una web.

3.7.5 WebStorm

Figura 3-9: Logotipo de WebStorm

WebStorm es una herramienta de desarrollo que ayuda a la implementación de una aplicación en Javascript. Con este entorno de desarrollo se implementará todo el código Javascript que sea necesario a la hora de crear la web para mostrar los datos.

3.7.6 Bash

Figura 3-10: Logotipo de Bash

Bash es un programa que proporciona la interpretación y ejecución de comandos mediante una consola en entornos Unix. Comúnmente se conoce a estos archivos scripts.

Se utilizará este programa para automatizar el proceso de arranque de los diferentes servicios y programas asociados para el correcto funcionamiento del sistema.

Adicionalmente se incluirá comandos extra en los scripts para crear procesos independientes que nos ayudarán a monitorizar la aplicación, generar copias de seguridad o en caso de carga de trabajo extrema o porcentaje bajo de espacio en el disco duro notificar o parar el sistema por completo o alguno de sus procesos o servicios.

3.7.7 Editor de texto

Figura 3-11: Logotipo de Sublime Text

Se utilizará un editor de texto, en este supuesto será sublime text, para codificar los ficheros PHP, los scripts necesarios y cualquier tipo de archivo que no sea necesario editarlo con un entorno de desarrollo específico debido a cambios puntuales o pequeñas modificaciones.

3.7.8 Solr

Figura 3-12: Logotipo de Solr

Apache Solr [19] es un motor de búsqueda (indexador) para almacenar datos y poder buscar después sobre ellos. Las principales ventajas de Solr son el bajo tiempo de búsqueda y la capacidad de ser escalable y tolerable a fallos.

Cuenta con funcionalidades muy deseables como un monitor para poder descubrir el origen de un posible fallo o contar con una amplia comunidad para resolución de dudas y preguntas.

Esta herramienta se usará para buscar sobre determinadas palabras que estén en las publicaciones

3.7.9 Tomcat

Figura 3-13: Logotipo de Tomcat

Tomcat es un software libre que tiene como característica el poder de contener servlets. En él se añadirá la funcionalidad añadiendo el código Java correspondiente para que sea capaz de mantener la conexión con Twitter y obtener la información necesaria.

3.7.10 Spark

Figura 3-14: Logotipo de Spark

Spark es un framework de software libre para poder distribuir el trabajo entre diversas máquinas para obtener un mayor rendimiento. Se utilizará este software en este sistema para poder conseguir un mejor rendimiento al poder contar con un conjunto de máquinas trabajando a la vez.

También se podrá satisfacer de manera óptima las necesidades de cómputo en cualquier periodo de tiempo. Por ejemplo si en un futuro se necesitase mayor capacidad de procesamiento por un crecimiento en el volumen de datos sencillamente se aumentaría el número de máquinas en el clúster.

3.7.11 MongoDB

Figura 3-15: Logotipo de MongoDB

MongoDB es un tipo de base de datos NoSQL basada en documentos. La forma de guardar los datos difiere de las habituales bases de datos relacionales. MongoDB guarda estructuras BSON (tipo de dato similar a JSON). Brinda escalabilidad, rendimiento y gran disponibilidad.

3.7.12 Ubuntu

Figura 3-16: Logotipo de Ubuntu

El sistema operativo sobre el que se desarrollará y en el que se implantará el sistema será Ubuntu.

La distribución que se utilizará para el desarrollo será la versión Desktop mientras que la versión que se utilizará para la implantación del sistema en las diferentes máquinas del clúster será la Server.

3.7.13 Navegador Web

Figura 3-17: Logotipo de Chrome y Firefox

Para testear y verificar que la parte web del sistema funciona correctamente, se utilizarán varios navegadores para garantizar que el funcionamiento es igual en ambos.

En este caso se utilizarán los navegadores web más populares, Chrome y Firefox con sus respectivas extensiones para desarrolladores.

3.7.14 SeleniumHQ

Figura 3-18: Logotipo de SeleniumHQ

Para automatizar las pruebas de la web lo máximo posible se usará una extensión para Firefox, SeleniumHQ.

Utilizando este software de automatización se podrá validar los desarrollos incrementales en la web, aunque habrá que repetirlos en el otro navegador manualmente (Chrome)

4 Diseño

En esta parte del documento se abarcará el diseño establecido y seguido para la posterior implementación del sistema.

En este capítulo se expondrán los inicios del proyecto, su arquitectura así como cada uno de sus componentes y el trabajo que desempeña cada uno.

4.1 Inicios del proyecto

El proyecto que se desea desarrollar es un proyecto financiado por el Directorado General Europeo para Asuntos Internos en el marco del Programa para la Prevención de/y Lucha contra el Crimen en 2012.

En 2014 se empiezan a producir las primeras reuniones del consorcio en las que finalmente forman parte HI Iberia como líder, la colaboración de la Policía Municipal de Madrid como usuario final y la Universidad de Middlesex (Reino Unido) como experto en asuntos éticos y legales.

Durante los primeros meses se lleva a cabo la revisión de los objetivos, resultados e innovaciones al igual que se empieza con una captación de los requisitos de la Policía Municipal como usuario final para la elaboración de los casos de uso que se usarán para posteriores pruebas.

En 2015 se presentan los primeros resultados de una búsqueda de información en redes sociales a través de una web temporal que no estaba basada en la tecnología que se iba a terminar realizando (MeteorJs). En este mismo año se focalizaron los casos de uso y se presentó la arquitectura diseñada que se irá implementando.

En este periodo de tiempo es cuando el autor de este TFG participa en este proyecto, centrándose en la implementación y diseño de la parte web, construida a partir de cero, aunque también ayudará en la implementación de otras partes de la aplicación.

4.2 Arquitectura del sistema

En este apartado se mostrará mediante una imagen ilustrativa la arquitectura del sistema y de qué elementos se compone para tener un enfoque general de su extensión antes de comenzar a ver el funcionamiento de cada elemento por separado.

También se podrá apreciar la comunicación entre cada uno de los módulos y ver el camino que siguen los datos desde el momento de su recepción hasta su procesado y almacenado en la base de datos.

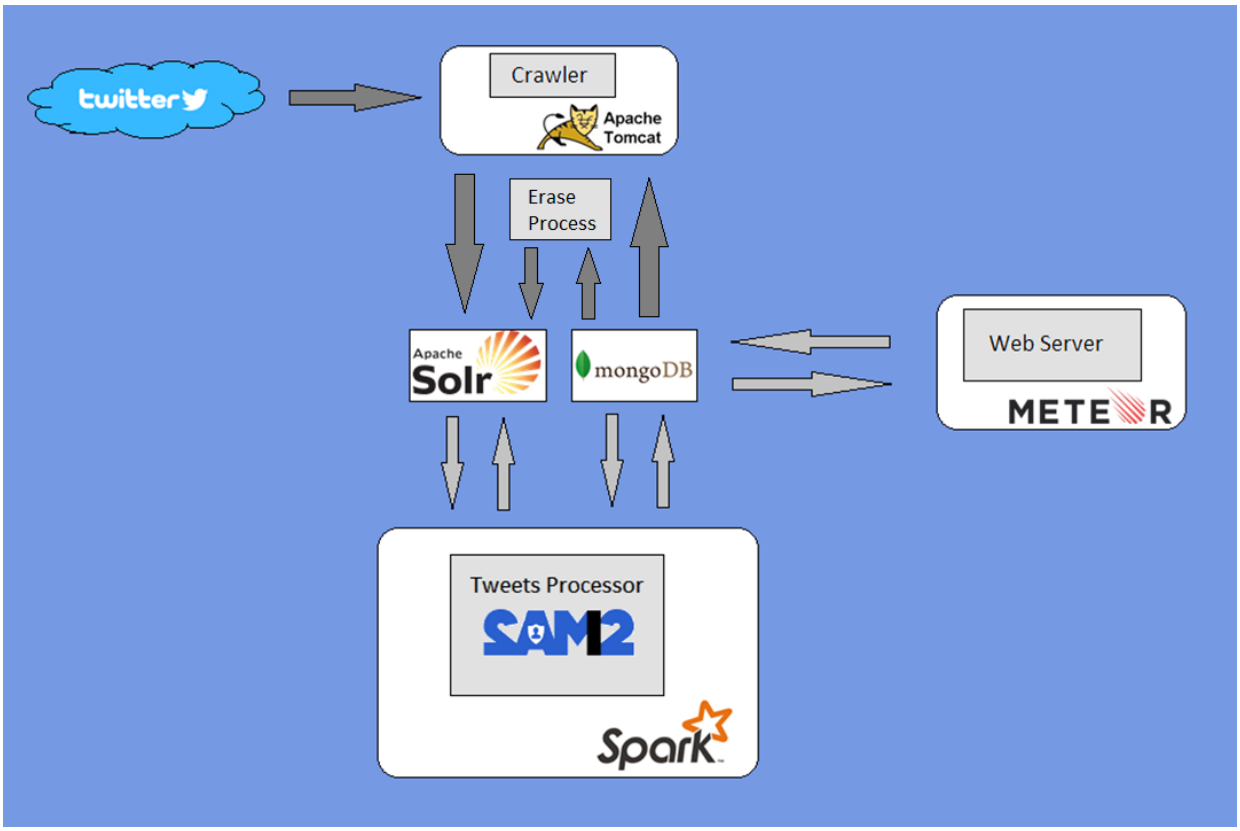

Figura 4-1: Arquitectura del sistema

Como se puede apreciar de la imagen anterior, el sistema está formado por seis módulos esenciales.

4.2.1 Twitter streaming API

Primeramente, el primer paso a elaborar es solicitar a Twitter publicaciones. Las publicaciones que se solicitarán, no serán todos los tuits de un usuario o grupo de usuarios en ni por ejemplo publicaciones de hace meses o años ya que no serían relevantes para el objetivo de evitar y prevenir o evitar contenido o actividades ilegales.

Las publicaciones que se solicitaran serán los tuits que se escriban en el mismo momento de la solicitud, es decir obtener las publicaciones que el usuario publique en tiempo real.

Otro condicionante es la ubicación de estos usuarios y su localización de cuando se publica un tuit. No sería relevante ni productivo procesar tuits de otro país ya que muy posiblemente no se encontrarán en nuestro idioma ni serán de utilidad para los usuarios finales.

Con todos los requisitos anteriores, Twitter proporciona una solución. La solución es su streaming API. En ella se pueden solicitar las publicaciones (siempre que sean públicas) de todos los usuarios en tiempo real además de poder realizar un filtrado antes de que sean devueltas mediante un bounding box.

Parameter value	Tracks Tweets from...
-122.75,36.8,-121.75,37.8	San Francisco
-74,40,-73,41	New York City
-122.75,36.8,-121.75,37.8,-74,40,-73,41	San Francisco OR New York City

Figura 4-2: Ejemplo de bounding box de Twitter

De esta manera para la solicitud se introducirán las coordenadas GPS de España y pasarlo como parámetro al utilizar la API. La imagen anterior muestra las coordenadas que habría que enviar como parámetro para las diferentes ciudades y la siguiente un ejemplo de petición de tuits de la ciudad de San Francisco.

```
https://stream.twitter.com/1.1/statuses/filter.json?
delimited=length&locations=-122.75,36.8,-121.75,37.8
:

HTTP/1.1 200 OK
Content-Type: application/json
Transfer-Encoding: chunked
```

Figura 4-3: Ejemplo de petición a twitter

La labor de mantener esta conexión y generarla correrá a cargo del siguiente componente, el crawler.

4.2.2 Crawler

El crawler utilizará Apache Tomcat para su correcto funcionamiento. En él se añadirá un servicio extra en forma de servlet para que realice y mantenga la conexión con Twitter abierta para obtener las publicaciones que la red social nos vaya enviando para posteriormente almacenarlas en nuestra base de datos e indexador en el formato adecuado.

En el momento que se está en posesión de una publicación válida proporcionada por Twitter, se almacenará en la base de datos MongoDB y en el indexador aunque con una pequeña modificación.

La modificación a la que se someterá cada publicación será a preservar el anonimato de la persona la cual escribió el tuit. Para esta labor, se utilizará un algoritmo de hash con el condicionante de que no puede haber colisiones en el valor hash resultante y éste preferiblemente no debería tener una longitud fija muy elevada, ya que al mostrar los resultados de las búsquedas realizadas por el usuario será este valor el que verá el usuario como el autor de la publicación. Para ello se utilizará una versión con suficientes bits para minimizar al máximo las colisiones posibles o incluso eliminarlas y adaptarnos a la preferencia de un nombre excesivamente largo.

4.2.3 MongoDB

La base de datos NoSQL realiza dos funciones diferentes, cada una en procesos independientes. La primera, es la inserción de datos, que puede deberse a insertar las publicaciones recibidas por el crawler directamente o la inserción de los perfiles de búsqueda creados en la web. La segunda es la consulta de datos al interactuar con la web para su visualización.

4.2.3.1 Inserción

En la inserción de las publicaciones, éstas se almacenan anonimizadas por el crawler para su posterior procesado mediante otro proceso más exhaustivo de procesador de texto que se mencionará más adelante.

Otra labor que corre a cargo de esta base de datos es almacenar los perfiles de búsqueda creados en la web. Dentro de cada perfil de búsqueda se encuentra también su propia red neuronal y las publicaciones asociadas a cada perfil que haya recabado el procesador de publicaciones para su posterior consulta a través de la web.

4.2.3.2 Consulta

La consulta a esta base de datos puede realizarse de dos formas, y como se exponía anteriormente, también de forma de procesos independiente.

La primera consulta puede ser producida por el procesador de publicaciones, para acceder a la publicación original recibida de Twitter pero anonimizada para procesarla mediante un análisis más exhaustivo de texto.

El otro tipo de consulta es a través de la web. En ella se realizan consultas seleccionando un área de acción, un rango de tiempo determinado y un perfil de búsqueda. El resultado de este filtro sobre las publicaciones ya filtradas por el procesador de publicaciones serán altamente relevantes y de gran interés.

4.2.4 Solr

Solr también tiene dos procesos independientes como MongoDB, inserción y consulta.

4.2.4.1 Inserción

En Solr también se insertan las publicaciones recibidas por el crawler, pero antes de ser guardadas siguen un proceso para incluir más información sobre las palabras de la publicación. Este proceso se añade al proceso que ya usa solr para la inserción de información, es decir, se extiende.

El proceso propio que se añadirá a solr será buscar sinónimos para cada palabra en el texto y derivar el verbo para dejarlo en infinitivo. Para esta labor se utilizan bases de datos con sinónimos en castellano. De esta manera, se podrán encontrar publicaciones en una búsqueda que no coincidan sus palabras clave gracias al uso de los sinónimos.

4.2.4.2 Consulta

Para poder procesar esta gran cantidad de publicaciones primeramente se realiza un primer filtrado para obtener contenido relevante para un posterior análisis más en profundidad.

Este primer filtrado se realiza sobre el contenido de la publicación, es decir, de cada una de las palabras que está compuesto el mensaje. Mediante una consulta de Solr con palabras clave predefinidas en el perfil de la búsqueda. En la siguiente imagen se puede apreciar el uso de una consulta muy simple a Solr.

http://localhost:8983/solr/query?debug=query&q=hello


```
{
  "responseHeader":{
    "status":0,
    "QTime":0,
    "params":{
      "q":"hello",
      "debug":"query"}},
  "response":{"numFound":0,"start":0,"docs":[]
  },
  "debug":{
    "rawquerystring":"hello",
    "querystring":"hello",
    "parsedquery":"text:hello",
    "parsedquery_toString":"text:hello",
    "QParser":"LuceneQParser"}}
```


Figura 4-4: Ejemplo de consulta Solr

En el campo “q” es donde se añaden los parámetros con los que se desea realizar la consulta. En este caso serán las palabras que formen el perfil de búsqueda correspondiente creado previamente.

Esta consulta a Solr la realiza el procesador de publicaciones. Más adelante se explicará en detalle cómo funciona este procesador y que conexión realiza con Solr y otros componentes.

4.2.5 Procesador de publicaciones

El procesador de publicaciones es el corazón del sistema. Es cierto que sin los demás componentes no podría funcionar correctamente pero es el órgano vital para poder alcanzar el objetivo de este proyecto.

Este componente analiza cada publicación que Solr le reporta y realiza un análisis semántico de las frases para intentar reconocer cada elemento de la frase y poder inferir si ese elemento es un artículo, complemento circunstancial de lugar, de tiempo…

También es el encargado de utilizar la red neuronal para predecir las publicaciones resultado. A medida que procesa publicaciones y obteniendo una retroalimentación del usuario puede “aprender” y mejorar las posteriores búsquedas o enfocarse en una determinada dirección de búsqueda.

4.2.5.1 Funcionamiento

El funcionamiento del procesador de publicaciones es cíclico, es decir, cada cierto periodo de tiempo se vuelve a ejecutar la labor de este componente.

Este componente empieza su labor obteniendo los perfiles de búsqueda que hay creados hasta el momento. Para cada uno de ellos obtiene las palabras clave y lanzará una consulta a Solr que le reportará las publicaciones que superen ese filtro de palabras y unas restricciones de tiempo.

Para cada publicación retornada por Solr se le analiza semánticamente el texto para encontrar elementos sintácticos en la frase tales como verbo, complemento directo, complemento circunstancial de lugar…

De esta forma, se pueden inferir con este comportamiento gracias al analizador semántico de texto nombre de entidades asociadas en las que en un determinado momento del tiempo se vuelven tendencia en la red social y se escribe sobre ellas.

Una vez realizado este análisis profundo del texto los resultados se almacenan en la base de datos MongoDB para las búsquedas desde la página web.

4.2.6 Servidor web

El servidor web tiene la función de conectar al sistema de procesado de publicaciones con el usuario para que pueda visualizar el análisis de los textos y lograr sacar partido a esos datos. Para ello, se utilizan dos funcionalidades principales que son la creación de un perfil de búsqueda y la consulta o búsqueda en la web haciendo uso de esos perfiles.

4.2.6.1 Perfil de búsqueda

El perfil de búsqueda se creará a través de la web de manera intuitiva para el usuario y en él se añadirán palabras clave además de otra información extra, como podría ser el tiempo verbal que se desea buscar, o una entidad o lugar en concreto. Con esto se mejorará la precisión e incidir en publicaciones más específicas.

Una vez que el perfil se ha creado satisfactoriamente se le asigna una red neuronal inicial. Esta red neuronal es propia del perfil de búsqueda e irá evolucionando con el paso del tiempo con la información que reciba del usuario cuando evalúe si los resultados obtenidos en las búsquedas de dicho perfil son relevantes o no. De esta manera se podrá afirmar que cuantas más publicaciones se evalúen, ya sea que se marquen como relevantes o no, el trabajo de búsqueda será mejor y más eficiente en el futuro.

También en este momento, el procesador de publicaciones -que está funcionando periódicamente- empieza a procesar publicaciones para este perfil.

4.2.6.2 Búsqueda

A continuación se tratará el proceso de búsqueda. Este proceso involucra al servidor web y a la base de datos MongoDB.

En primer lugar, para hacer uso de la búsqueda se seleccionará un perfil de búsqueda previamente creado, la ubicación y el periodo de tiempo que se desea filtrar.

Con estos parámetros se filtrarán las publicaciones que el procesador de publicaciones ha reportado y se mostrarán por pantalla con un aspecto similar al de la propia red social pero con mayor información y funcionalidad.

La información que se ampliará será el número de publicaciones analizadas, el número de publicaciones resultado, así como el número de hashtags y menciones a otras personas sobre las publicaciones resultado.

Como funcionalidades extra:

- Se aportarán un filtrado sobre las publicaciones resultado mediante tres listas de datos. Las listas de datos incluyen los hashtags, las menciones y las entidades respectivamente. Por ejemplo, se podrá seleccionar uno o varios hashtags o/y menciones o/y entidades y las publicaciones que se verán deberán cumplir ese filtrado.

- Se podrá seleccionar una publicación resultado y obtener su análisis social, que es el árbol de tuits (respuestas) y los usuarios relacionados con esas respuestas.

- Se podrá observar la ubicación de la publicación al seleccionarla. Esta ubicación tendrá como preferencia mostrar la ubicación real de donde se escribió el tuit, pero no siempre es posible con lo que se mostraría la ubicación que seleccionó el usuario al crear su cuenta en la red social.

4.2.7 Proceso de borrado

El proceso de borrado es un proceso independiente el cual periódicamente realiza peticiones a MongoDB y Solr de borrado de publicaciones con fecha de publicación superior a un mes o próxima a este periodo.

Este proceso se realiza para proteger la privacidad de los usuarios y también para no desbordar la capacidad de los discos duros de las máquinas.

4.2.8 Escalabilidad del sistema

El trabajo a realizar para esta aplicación puede poner en serios problemas a una sola máquina si desea realizar todo este trabajo. Para ello, y no tener que contar con un equipo de grandes prestaciones se utilizará un clúster. Con esta medida se podrá tener equipos más económicos y si en un cierto periodo de tiempo el procesado necesita mayor capacidad de cómputo, no haría falta reemplazar las máquinas existentes, con la inclusión de más maquinas se podría solventar el problema.

Para obtener esta funcionalidad se utilizará Spark. Con este programa de software libre se tendrá un nodo central que alojará todos los componentes del sistema menos el procesador de publicaciones. De esta manera el trabajo en paralelo del procesador de publicaciones hará que aumente el rendimiento.

Esta modalidad tiene una modificación en la base de datos MongoDB para evitar que pudiese ser un cuello de botella o un único punto de fallo ya que todos los nodos esclavos acceden a una única base de datos. Para solventar este inconveniente se usará una funcionalidad que brinda la propia base de datos que es el uso de réplica set.

Esta modalidad disponible tiene un principio similar a la configuración de discos duros en raid 5, es decir, cada esclavo tiene una porción de la base de datos considerada primaria y el resto secundarias. Con esta configuración se mejorará la disponibilidad y permanencia de los datos en caso de que uno de los nodos deje de funcionar.

4.2.9 Ciclo de vida de una publicación

Una vez descritos cada uno de los componentes y sus funcionalidades de manera separada, ahora se expondrá el flujo o ciclo de vida que tiene una publicación desde el momento que es escrita en la red social hasta el momento que llega a ser mostrada en la página vez mediante una búsqueda. Los procesos que se tratarán a continuación son completamente independientes y están continuamente ejecutándose.

4.2.9.1 Proceso de recolección de publicaciones

Este proceso se inicia cuando un usuario escribe un tuit y ese tuit cumple las condiciones de localización necesarias para que Twitter lo envíe mediante el uso del crawler.

Una vez el crawler lo anonimiza lo inserta en la base de datos MongoDB y en el indexador Solr con la peculiaridad de que en Solr se inserta información extra como sinónimos y derivaciones del verbo.

En este momento las publicaciones estarán a la espera de que el proceso de análisis haga uso de ellas.

4.2.9.2 Proceso de análisis

Para que el proceso de análisis comience debe haber al menos un perfil de búsqueda creado.

Una vez que existe al menos un perfil de búsqueda el sistema periódicamente realiza consultas a Solr con las palabras clave de cada perfil y buscando en un periodo de tiempo determinado. Este periodo de tiempo se configura en el perfil de búsqueda para cuando se cree busque publicaciones con una antigüedad preestablecida, por ejemplo un meses.

En cada ciclo de ejecución irá avanzando ese periodo hasta llegar a la actualidad y mantenerse actualizado. Es decir, primeramente hará una consulta de publicaciones con una antigüedad de entre 30 y 25 días de antigüedad, después entre 25 y 20 días y así sucesivamente.

Para cada publicación que sea retornada por Solr se le aplicará el análisis semántico para obtener más información sobre la frase, especialmente para poder inferir entidades.

Una vez que se ha procesado el tuit se entrena en la red neuronal para indicarle a esa publicación un valor de resultado que será almacenado.

4.2.9.3 Proceso de búsqueda

El proceso de búsqueda en la web únicamente debe realizar consultas a la base de datos de MongoDB sobre las publicaciones asignadas al perfil de búsqueda.

Para la consulta será necesario introducir en la web el perfil de búsqueda, la zona de acción y el rango de tiempo deseado.

Una vez introducidos los parámetros anteriores en la web se obtienen los resultados. Estos resultados se obtienen de una consulta a MongoDB de las publicaciones que han pasado el análisis semántico y la red neuronal. También deben pasar los filtros de área geográfica y rango de tiempo.

Por otra parte, para mejorar la red neuronal en el camino de búsqueda que el usuario desee, puede evaluar cada publicación que ha obtenido como resultado. Este cambio de publicación a relevante o no relevante tiene un efecto directo sobre la red neuronal, con lo que la modifica para sucesivos tuits que se procesen.

4.2.9.4 Proceso de destrucción

Como se explicó anteriormente y sólo a modo de sintetizar en esta sección todos los procesos se vuelve a citar este proceso.

El proceso de destrucción es otro proceso más independiente que se ejecuta en el nodo máster del clúster y periódicamente solicita a MongoDB y Solr la eliminación de publicaciones que han superado una antigüedad desde su publicación de aproximadamente un mes.

5 Desarrollo

En este capítulo se expondrá la etapa de desarrollo de este sistema. Esto representa la metodología de trabajo adoptada, el sistema de control de versiones así como se expondrá el mayor reto que ha supuesto el desarrollo de este sistema al igual que cómo se afrontó la inclusión de nuevas funcionalidades.

Este apartado recoge a su vez los mecanismos creados para que el desarrollo sea lo más llevadero posible y con mente en el mantenimiento futuro. En este marco se encuentran los scripts creados, los registros almacenados, la monitorización, respaldo y restauración de información así como la documentación.

5.1 Subversion

El desarrollo comienza estableciendo un sistema de control de versiones para el código que se irá desarrollando. Con esta funcionalidad se consigue gestionar el desarrollo de cada programador y poder revertir cambios si en algún momento durante el desarrollo si se ha cometido un error en una versión superior.

También conlleva una serie de beneficios adicionales. Primordialmente tener el código salvaguardado por si en la máquina de desarrollo podría ocurrir algún percance como su pérdida, infección por un virus o rotura.

Proporciona un avance y sincronización entre los miembros del equipo de trabajo, ya que diariamente se solicitan los cambios para en la medida de lo posible intentar integrar módulos lo más rápidamente o comenzar desarrollos que dependan de la codificación de otra persona.

Por último, el poder incluir un mensaje en cada envío de código al servidor explicando los cambios realizados y funcionalidades completadas hace que la vuelta atrás sea lo más precisa posible si en algún momento una versión antigua resulta de utilidad. También de esta manera se puede llevar un control de la productividad de cada persona y la dificultad del trabajo asignado para poder ayudar en momentos que la carga de trabajo sea muy elevada o la persona asignada se encuentre con imprevistos o situaciones no contempladas en el diseño.

Para gestionar este control de versiones de código se usará un servidor ubicado en HI Iberia destinado para el control de versiones en los múltiples proyectos en desarrollo.

Como cliente a utilizar para subir y descargar los cambios realizados por cada uno de los integrantes del equipo de trabajo se utilizará el plugin subversive para eclipse para el desarrollo en Java y la funcionalidad nativa en WebStorm para el desarrollo completo de la parte web.

Para la subida de documentos tales como documentación, archivos de configuración o scripts se utilizará el software SmartSVN o alguno de los clientes anteriormente descritos.

5.2 Metodología de trabajo

La metodología de trabajo adoptada para este desarrollo es scrum pero con ciertas modificaciones.

Se modifica el plazo entre reuniones, en vez de ser diariamente se realiza una reunión cada tres o cuatro días aproximadamente. Esta modificación hace que se desempeñase más tiempo desarrollando y escribiendo código. Aunque, eventualmente si algún integrante del equipo de trabajo detectaba algún percance que perjudicase su desarrollo hasta la siguiente reunión se adelantaba dicha reunión o se convocaba una de urgencia.

En cada reunión, con una duración inferior a 45 minutos, cada integrante exponía el trabajo que había realizado desde la última reunión y sus expectativas para la siguiente o siguientes. En este momento si había algo funcional los demás integrantes podían ver una demostración o realizarla ellos mismos con la finalidad de poder sugerir alguna mejora significativa o detectar errores.

Al exponer el trabajo que se iba a realizar los próximos días, se debatía rápidamente como se podría dirigir el desarrollo o incluso el jefe de proyecto indicaba requisitos o preferencias que el usuario final le había indicado en la sucesión de mensajes intercambiados.

Este funcionamiento aseguraba casi a la perfección que el resultado final, tanto estético como a funcionalidad iba a encajar con lo esperado por el cliente.

En cada iteración en el desarrollo se obtenía un prototipo que se enviaba al usuario final para que pudiesen probarlo. Aunque no todas las funcionalidades funcionasen completamente, este paso resultaba crucial para el equipo, ya que pequeños errores o mejoras para el filtrado o interactuación eran subsanados al instante.

5.3 Scripts

Los scripts son de gran ayuda cuando periódicamente se tiene que realizar acciones repetitivas que se podrían automatizar en un solo fichero. También ayuda a no cometer fallos por no introducir la secuencia correcta de comandos por la terminal del sistema que conllevarían a deshacer la secuencia de operaciones actual.

Un punto clave más a favor es la capacidad de abstraer y no tener que recordar cada una de las acciones que se debe realizar. Así como la facilidad de configuración o de relanzado de servicios y procesos después de un reinicio.

En este desarrollo se crearon scripts incrementales, es decir, se generaron uno por cada servicio asociado para poder lanzar o parar cada servicio con su configuración correspondiente. Posteriormente se integraron varios scripts de servicios para lanzar o parar módulos completos y por último se creó un script padre que posee la cualidad de levantar o detener todo el sistema.

5.4 Logs

Este sistema se compone de múltiples módulos y servicios gestionados por diversos programas externos de software libre. En la etapa de implementación y desarrollo es cuando más errores surgen, tanto como propios como motivados por herramientas externas (falta de recursos, errores de dependencias, etc.)

Como todos los procesos de este sistema se ejecutan en segundo plano permanentemente y no hay una terminal o salida por pantalla para mostrar los diferentes errores o información asociada a la ejecución es de vital importancia contar con un sistema de logs, no sólo para encontrar el punto de fallo en el caso de error fatal, si no también llevar un control de cómo transcurre la ejecución de cada módulo.

En este sistema será muy importante saber cómo se procesa cada publicación por si en algún momento el procesamiento o el valor o la modificación de la red neuronal no está funcionando según lo esperado.

Con un sistema de logs implementado por niveles y con un buen reporte hacia un fichero proporcionará una corrección muchísimo más rápida y eficaz frente a un error o comportamiento anómalo no deseable.

Por todo lo anterior, este sistema se compone de diferentes registros:

- Un log de Tomcat. En él incluimos únicamente los identificadores de las publicaciones que se reciben y se almacenan en la base de datos y el indexador.
- Un registro de MongoDB. En él se almacena las trazas de ejecución cuando hay un error y las consultas ejecutadas.
- Un log de Solr. En él se almacenaran los errores producidos con una descripción para poder solventarlos así como el proceso que sufre cada publicación al pasar por los filtrados propios en el proceso de indexación.
- Un registro del procesador de publicaciones. En él se almacenan los cambios producidos en una publicación después de que se analice mediante el analizador semántico y el valor que proporciona la red neuronal después de pasar la publicación analizada.
- Un log del espacio libre disponible. Más tarde, cuando se vea la monitorización se tratará el porqué de la monitorización de las máquinas. En este registro se almacenará cada pocos minutos la capacidad de espacio libre de cada máquina.

5.5 Restauración y volcado de información

Es preciso contar con una funcionalidad que pueda restaurar la información a un determinado momento del tiempo en el que el sistema era estable y los datos procesados y analizados eran correctos. Para ello también es necesario un sistema de respaldo de información, la cual será utilizada por el proceso de restauración.

Este proceso es altamente recomendable ya que permite una recuperación parcial de los datos gestionados por el sistema.

En este sistema la funcionalidad de respaldo y restauración de datos se hizo sobre la red neuronal, ya que puede sufrir modificaciones por error de los usuarios en una evaluación con bastante probabilidad. Para esta funcionalidad, antes de modificar la red neuronal se guarda una copia y se procede a su modificación. Esta copia queda registrada en el perfil de búsqueda correspondiente.

Con esta solución, aparte de poder revertir cualquier modificación de la red neuronal, se podrá determinar el lugar donde se cometió un error para que el funcionamiento no sea el esperado o cómo ha ido evolucionando la red neuronal para intentar optimizar y mejorar su funcionamiento.

5.6 Monitorización

Debido a que este sistema está en fase de desarrollo y dispone de un espacio reducido de capacidad de disco, se dispondrá de un proceso de monitorización del espacio disponible en disco de cada uno de los nodos esclavos del clúster.

Este proceso es un script. Se ejecuta en el nodo máster, y cada 10 minutos aproximadamente recoge el espacio disponible de cada máquina. Para que ninguna de las máquinas se quede sin memoria operativa y dejen de funcionar, al llegar al 95% de capacidad automáticamente detiene el streaming del crawler para dejar de recibir publicaciones.

5.7 Documentación

La documentación en cualquier proyecto de desarrollo de software es fundamental, dado que lo más común es que una vez finalizado el desarrollo la persona dedicada a su mantenimiento no es la misma que la que la desarrolló o incluso lo más probable es que aunque siendo la misma persona, ésta se haya olvidado de gran cantidad de funcionalidades y de cómo funcionaba cada componente o módulo.

Para paliar este resultado, desde la época de diseño de este sistema se ha ido documentando qué componentes se iban a utilizar y en qué tipo de configuración. Por ejemplo, la base de datos

MongoDB en modo de replicación. Este procedimiento no acaba sólo en la etapa de diseño sino que se ha ido manteniendo también en la etapa de implementación. Todo el código referente a cualquier módulo del sistema esta comentado.

Primeramente, arriba de cada fichero aparece el nombre, una breve descripción del comportamiento o funcionamiento y el autor o autores del mismo.

Para los archivos con extensión .java cada uno de los métodos contiene la suficiente información para conocer cómo usar cada método, con qué parámetros y que funcionamiento realiza. También este tipo de métodos están comentados para poder generar documentación mediante javadoc.

Para los métodos y funciones desarrollados para el servidor web, se sigue un procedimiento análogo como a los archivos en lenguaje java. Siguen el mismo formato de documentación.

Aún con este tipo de documentación, el código a veces puede ser poco claro al leerlo. Por este motivo, para no dejar la mínima duda sobre cualquier bloque de código, siempre hay un comentario explicativo en las secciones que han debido ser modificadas. Esto quiere decir que incluso el programador que ha estado desarrollando ha tenido dudas o dificultades para desarrollar y comprender el funcionamiento de ese bloque, motivo por el cual quedará explicado para que en el futuro no sembrar la mínima incertidumbre.

5.8 Página web

Para que el sistema sea amigable y de uso intuitivo de cara al usuario final, se incluye en el desarrollo la implementación de una página web que la labor que tendrá será la muestra de las publicaciones analizadas en la base de datos.

En este caso, como se expuso anteriormente se utilizó el framework MeteorJs. Este framework permite la implementación de una página web de manera muy sencilla y rápida aparte de proporcionar en manera de paquetes funcionalidades muy comunes y deseables en páginas web.

Para la implementación de esta parte del sistema se ha partido del diseño que proporciona Twitter pero ampliando su funcionalidad para optimizar y ayudar al usuario final en el uso específico que le va a dar.

En diversas partes del desarrollo del sistema ha habido complicaciones y contratiempos, pero es en esta parte, en la que el autor de este TFG considera que ha sido la más complicada del desarrollo, por lo cual se dispone a exponer este suceso.

La gran cantidad de publicaciones que almacena la base de datos y la restricción de un tiempo de respuesta casi nulo o de unos segundos para la muestra de por pantalla de las publicaciones resultados de una búsqueda supuso un gran reto.

El tiempo de espera desde que el usuario realizaba una búsqueda hasta que podía visualizar datos reales era del orden de minutos. Este funcionamiento no se podía tolerar, la visualización debía

ser prácticamente instantánea o como mucho esperar unos segundos. Aparte de la gran cantidad de tiempo que tardaba en mostrar la información, ésta era excesivamente grande. Se recibía para mostrar en pantalla miles de publicaciones.

Para solventar este problema se optó por usar el mismo procedimiento que proporciona Twiter, que es mostrar la información poco a poco partiendo de la más actual. Esto no era suficiente, puesto que este sistema debería ser capaz de filtrar por hashtags, menciones y/o entidades, por lo que se debe disponer de todos esos campos de filtrado de cada publicación. Y lo más importante, disponer de ellos en un tiempo que no comprometa la experiencia del usuario al utilizar la web. Para ello, se hizo uso de MapReduce, funcionalidad que proporciona MongoDB. Para poder usar este tipo de funcionalidad se tuvo que optimizar la información que se insertará al realizar este procedimiento. Contra menos datos se obtengan de la publicación para insertar en el proceso de MapReduce, se conseguirá un mayor rendimiento y menor uso de memoria.

Aún no han acabado los problemas relacionados a la página web. Al hacer uso de ella se verá que al navegar entre diversas páginas se perderá el cálculo de procesamiento de la base de datos ha hecho para mostrar los datos mediante paginación y ejecutar el MapReduce. Para ello se debe mantener de manera persistente los datos obtenidos entre las diferentes páginas.

Para solventar este nuevo inconveniente, y poder acceder a la página web desde la mayoría de los navegadores actuales, se opta por la utilización de sessionStorage, funcionalidad que proporcionan tanto los navegadores Chrome como Firefox y proporciona una capacidad de almacenaje más que necesaria.

5.9 Nuevos requisitos

Durante el desarrollo a veces suele ocurrir que el cliente solicite nuevas funcionalidades o cambios que en el diseño no se estudiaron o que durante la fase de desarrollo al ver y probar el prototipo piensan que sería deseable alguna funcionalidad extra.

Durante el desarrollo, el usuario final solicitará dos nuevas funcionalidades que le gustaría añadir al servidor web:

- Una página de estadísticas de la búsqueda. En ella se mostraría un resumen de la búsqueda realizada con tres nubes de tags. Una con los hashtags, otra con las menciones y la última con las entidades. Cada una de esas nubes deberá mostrar de manera clara y sencilla qué usuarios, menciones o hashtags se menciones más. Para ello, el tamaño de letra de las palabras con más ocurrencias será mayor. También se añadirá un mapa con todas las ubicaciones de los tuits para que el usuario conozca en qué lugares se escriben más publicaciones.

- Otra funcionalidad que se desea añadir es poder generar un informe sobre las publicaciones buscadas para en caso de que una publicación sea ofensiva o represente una amenaza pueda ser reportada para tomar las medidas legales oportunas.

Debido a estas nuevas funcionalidades, a los requisitos del informe (marca de agua, editable desde la web, y que incluya las publicaciones y las estadísticas) y viendo que es viable incluir esta funcionalidad, se elige realizar este informe sobre PHP ya que es el lenguaje que mejor se adapta a las solicitudes expuestas y mayor capacidad de mejora futura podrá soportar. Es por ello que se deberán incluir dos nuevas herramientas al sistema.

5.9.1 PHP

Figura 5-1: Logotipo de PHP

PHP es un lenguaje muy popular y extendido para el desarrollo web, el cual se utilizará de manera externa para generar informes en los que se podrá incluir publicaciones que el usuario final considere relevantes para sus posteriores acciones legales.

5.9.2 Apache

Figura 5-2: Logotipo de Apache

Apache es un servidor web HTTP de código abierto para sistemas operativos como Windows y Unix.

Se utilizará este servidor para alojar los ficheros PHP correspondientes para la generación externa de informes.

6 Integración, pruebas y resultados

En este capítulo se tratarán las cuestiones relacionadas con la integración de cada uno de los componentes así como las pruebas realizadas a cada módulo. Para terminar, se mostrará la parte visual de este desarrollo.

6.1 Integración

La integración de este desarrollo supone una dificultad añadida debido a la cantidad de módulos y componentes que lo forman, además de que este sistema cuenta con la capacidad de comunicarse con diferentes máquinas para realizar de forma paralela el análisis de publicaciones.

Primeramente para afrontar esta integración, se lleva a cabo una instalación de todas las tecnologías que se usan en las configuraciones necesarias descritas en capítulos anteriores. Como este sistema utiliza y se comunica con diferentes máquinas, se establecen en éstas relaciones de confianza para que se puedan comunicar entre ellas automáticamente y de manera segura.

Una vez probado que el software instalado y la comunicación entre las máquinas funcionan, se comienza la integración de los diferentes módulos de desarrollo. Para cerciorarse que dos módulos funcionan correctamente se utilizan pruebas de integración de módulos. Estas pruebas se automatizarán, en caso de los componentes Java a través de JUnit, y en caso del servidor web a través de SeleniumHQ. Utilizando estos programas y herramientas de test se podrá ejecutar en cada nueva integración todos los test anteriores y detectar si una nueva integración falla o si hace fallar a alguna de las ya añadidas.

6.2 Pruebas

Las pruebas siguen un proceso similar al de la integración pero con pruebas más exhaustivas y más a bajo nivel enfocadas a los algoritmos y bloques pequeños de código, así como a cada uno de los flujos de ejecución. Estas pruebas, son las denominadas pruebas de caja blanca. Mediante el uso de JUnit en Java y de funciones extras en el servidor web se logrará que el desarrollo esté lo más probado posible para evitar la mayoría de bugs.

Otras pruebas que se realizan automáticamente son las pruebas de caja negra, en la que no se prueban cada uno de los flujos que puede seguir el software, ya que éstas fueron realizadas anteriormente en las pruebas de caja blanca. En este tipo de prueba se intenta probar las funcionalidades de un módulo entero.

En la sección anterior se expuso la integración llevada a cabo y las pruebas que se realizaron, por lo que para terminar el bloque de pruebas sólo falta realizar las pruebas que no se pueden automatizar y con las que se validará el sistema completo.

La prueba final para ver si el desarrollo ha sido correcto y el sistema puede usarse por un usuario final que no tiene por qué tener conocimientos avanzados en informática no puede automatizarse. Para realizar esta prueba, se seleccionará una persona ajena al desarrollo dentro de la empresa para que intente interactuar tal y como lo haría el usuario final y recoger sus impresiones o puntos donde más dificultad encuentre con el motivo de mejorar. Una vez superadas estas pruebas, el sistema estará listo para que el usuario final lo pruebe.

6.3 Resultados

En esta sección se expondrá la parte visual de este desarrollo que es la página web. Las siguientes imágenes que contengan información relacionada con los usuarios de Twitter aparecerán difuminadas, dado que son datos reales.

6.3.1 Búsqueda

Al entrar en la página web, una vez superado la pantalla de ingreso en el sistema, nos encontramos con la búsqueda sobre publicaciones analizadas.

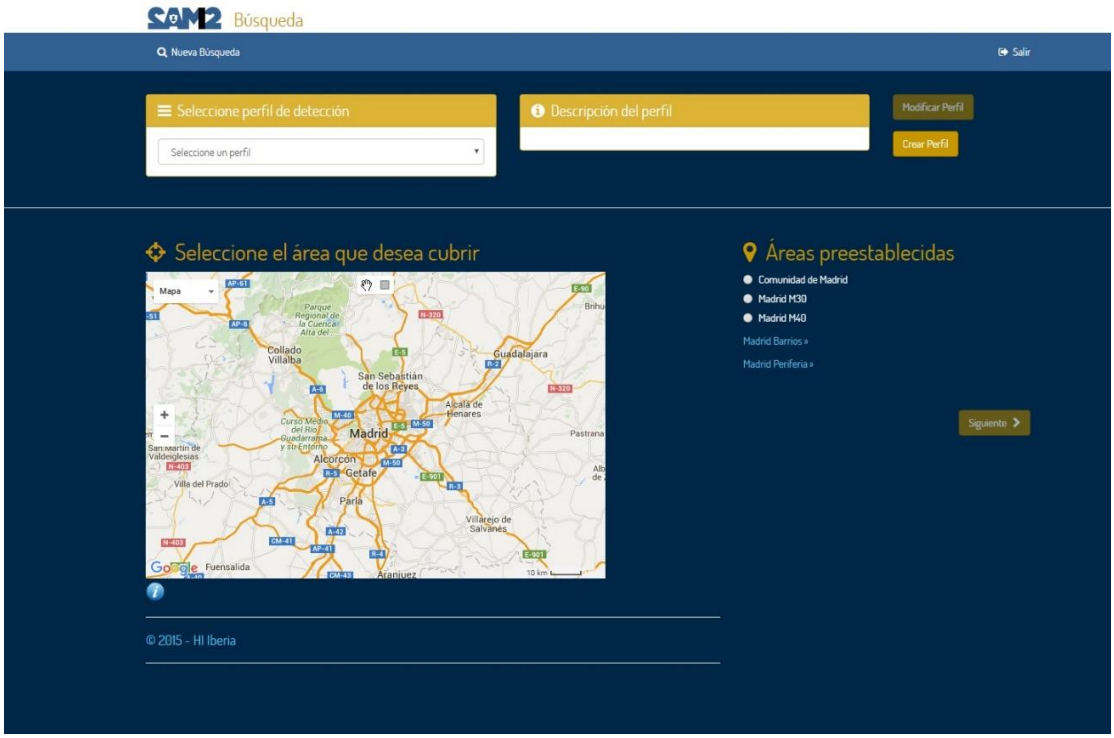

Figura 6-1: Búsqueda en SAMI2

En esta página y la siguiente se introducen el perfil de búsqueda así como unos parámetros que nos permitirán encontrar las publicaciones potencialmente deseadas. Estos parámetros son la localización, y el periodo de publicación.

6.3.2 Resultados de la búsqueda

A los pocos segundos se mostrarán las publicaciones que hayan superado el filtro de localización y rango de tiempo.

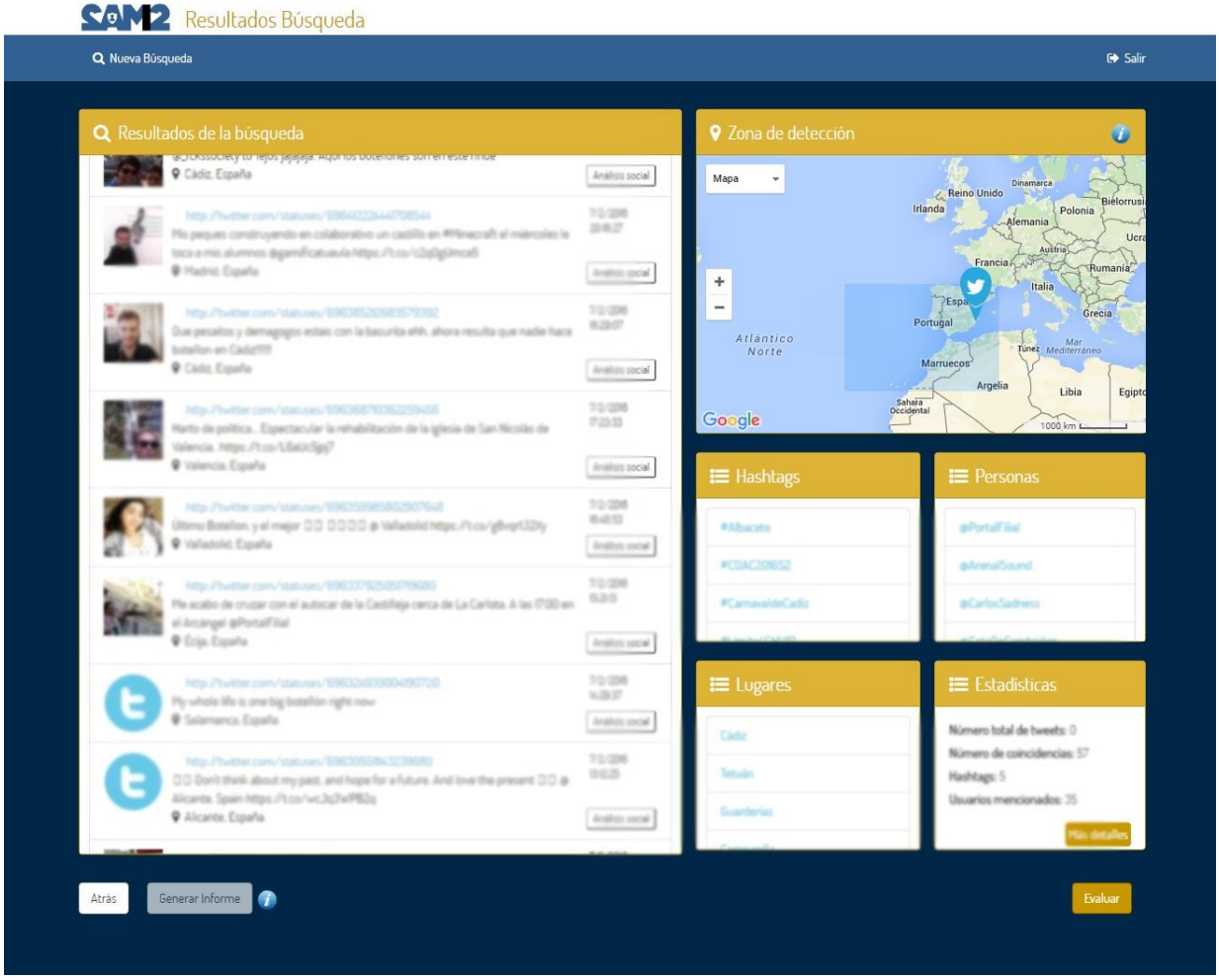

Figura 6-2: Resultados de búsqueda en SAMI2

En la imagen se muestra una lista de publicaciones ordenadas de manera cronológica, de manera que haciendo scroll se podrán ver otras más antiguas. En la parte derecha se observa un mapa de google maps, el cual muestra la ubicación de la publicación sobre la cual este el ratón del ordenador.

Más abajo aparecen cuatro recuadros. Los tres primeros se utilizan para filtrar las publicaciones de la lista como se observará más adelante y el cuarto recoge una serie de estadísticas sobre la búsqueda. Estas estadísticas son el número total de publicaciones analizadas, el número de coincidencias con los parámetros de búsqueda introducidos y el número de usuarios mencionados y hashtags.

6.3.3 Filtrar Resultados de la búsqueda

En la siguiente imagen se muestra un filtrado sobre las publicaciones obtenidas.

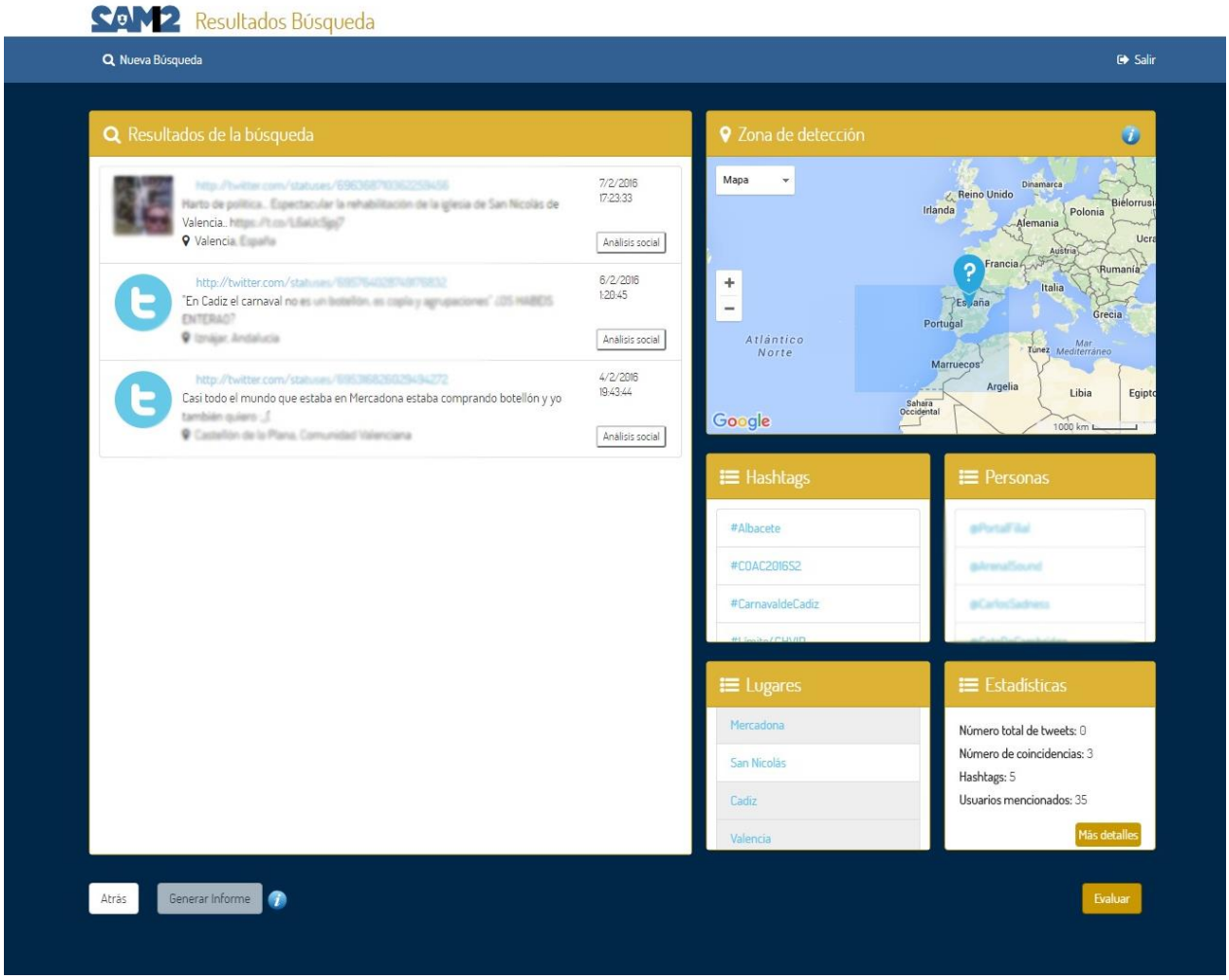

Figura 6-3: Filtrado de la búsqueda en SAMI2

En la imagen se puede apreciar el filtrado realizado sobre la búsqueda que se mostró en la imagen anterior. Se puede ver como los lugares, que son reconocidos gracias al analizador semántico del procesador de publicaciones son reconocidos en las publicaciones. También se puede hacer filtrado múltiple entra lugares, personas y hashtags indistintamente.

6.3.4 Evaluación de resultados

A continuación, se muestra la evaluación de resultados. A esta funcionalidad se llega desde los resultados de una búsqueda mediante un botón situado abajo a la derecha en la página.

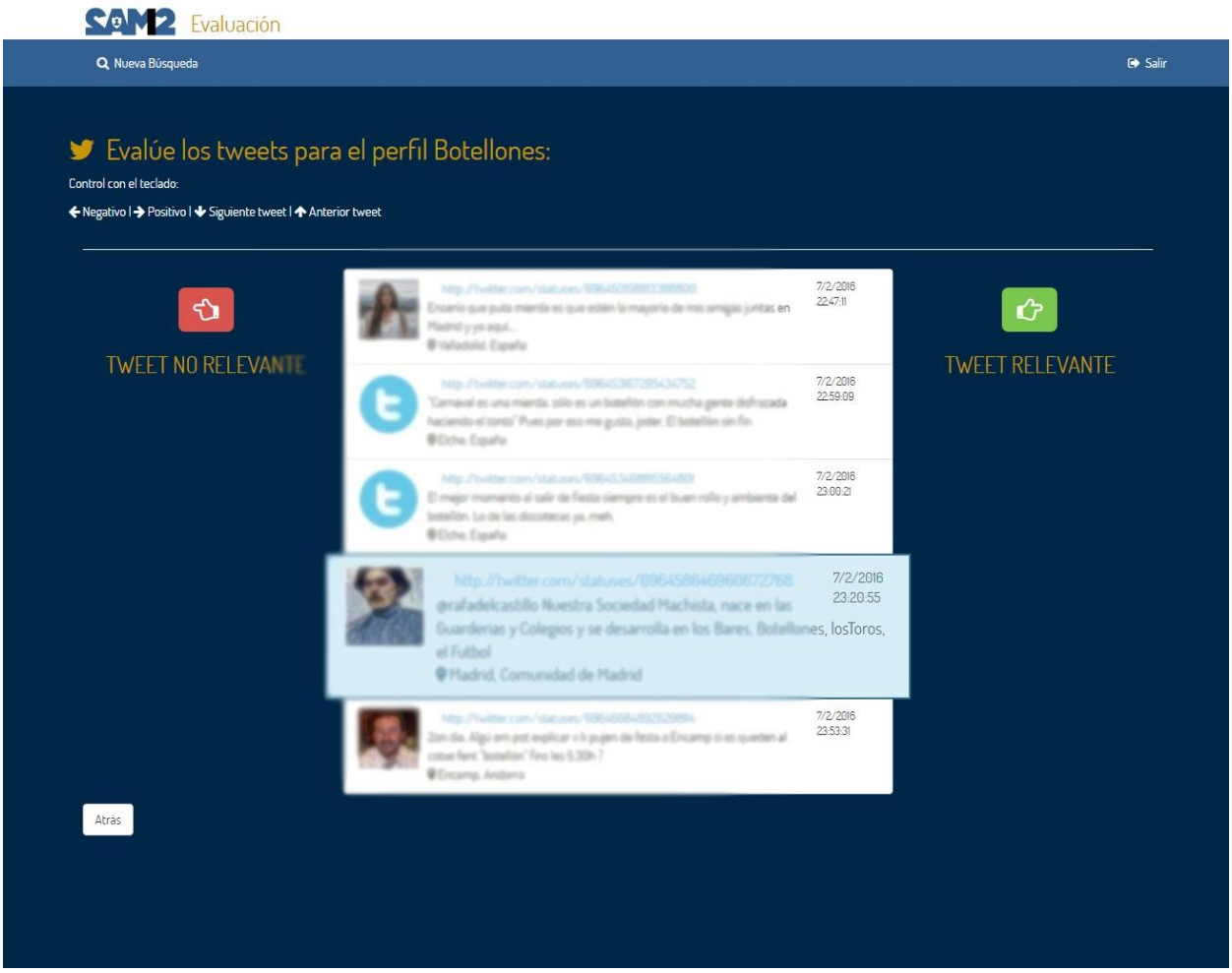

Figura 6-4: Evaluación de publicaciones en SAMI2

En esta página se puede indicar a la red neuronal que publicaciones son relevantes y cuáles no.

De esta forma intentará mejoras futuras búsquedas.

6.3.5 Análisis social

A continuación se mostrará el análisis social sobre una publicación. Para acceder a esta página se hará click sobre el botón próximo a la publicación en la lista de publicaciones al realizar la búsqueda.

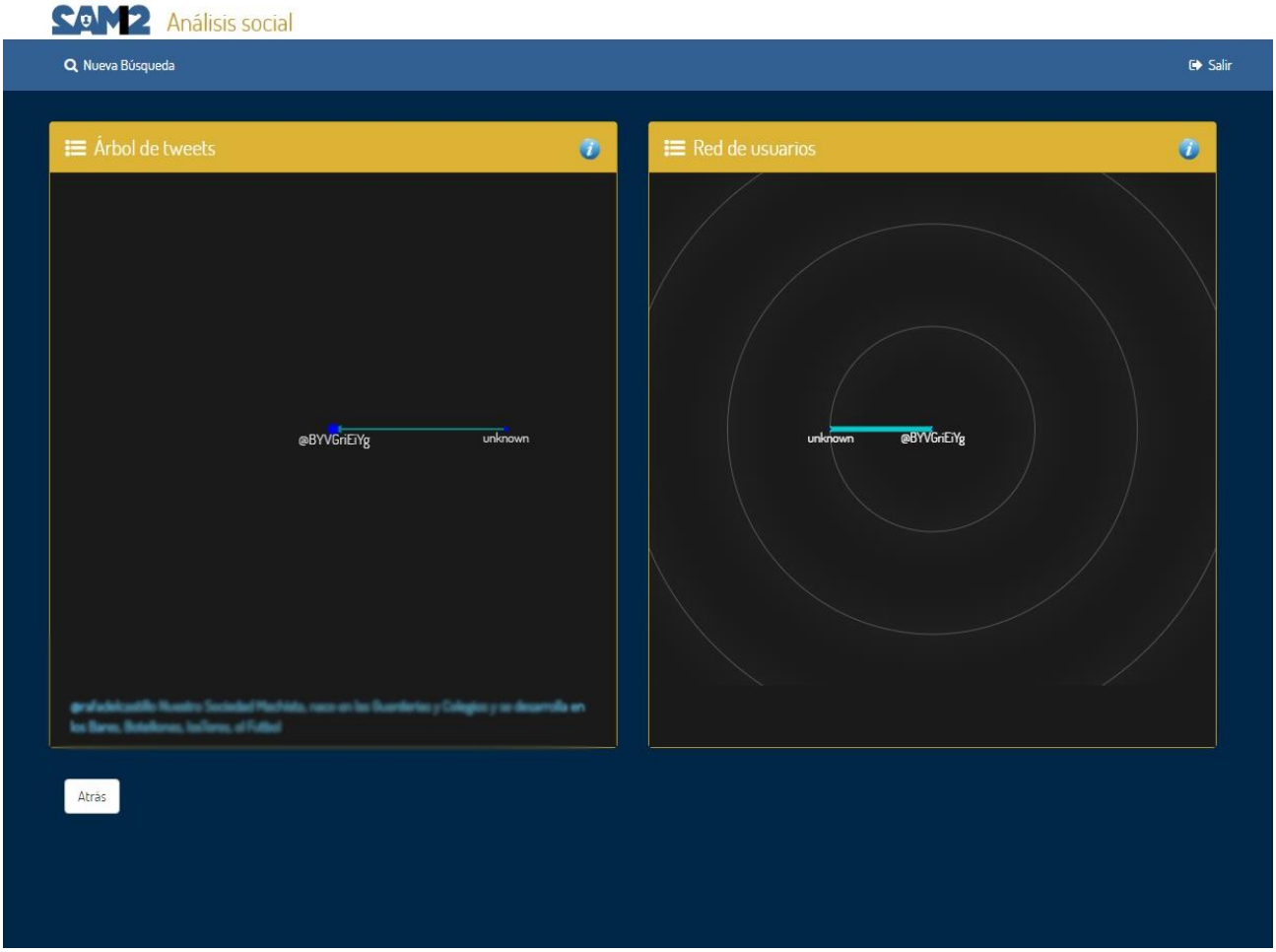

Figura 6-5: Análisis social en SAMI2

Como se puede ver gracias a la imagen anterior, se muestra un análisis social de la publicación.

Este análisis cuenta con un árbol de tuits y una red de usuarios, todo ello mostrado de manera visual e interactiva.

En el árbol de tuits aparece, aunque no siempre la publicación original y las respuestas que ha sufrido. No siempre esto es así debido a que una publicación de hace unos meses o años puede tener una contestación en la actualidad y como se puede ver en la imagen, esto es lo que ha ocurrido. La publicación de la que procede es “unknow”, es decir que no disponemos de ella en la base de datos.

En la parte derecha se muestra la red de usuarios que han interactuado con el autor de la publicación escribiendo una respuesta.

Estos gráficos son interactivos, es decir se podrá navegar por el árbol de tuits y ver la publicación de la otra persona, aunque en este caso mostraría un texto avisando de que la publicación no se encuentra en la base de datos.

De igual manera en la red de usuarios. Si ésta fuese tan grande que no se pudiera distinguir entre el nombre -anonimizado- de las personas que han interactuado, nos podríamos desplazar por cada uno de los nodos, mostrando todos los nombres.

6.3.6 Estadísticas

Para finalizar este capítulo, se mostrará una de las últimas funcionalidades añadidas al sistema. Las estadísticas de la búsqueda.

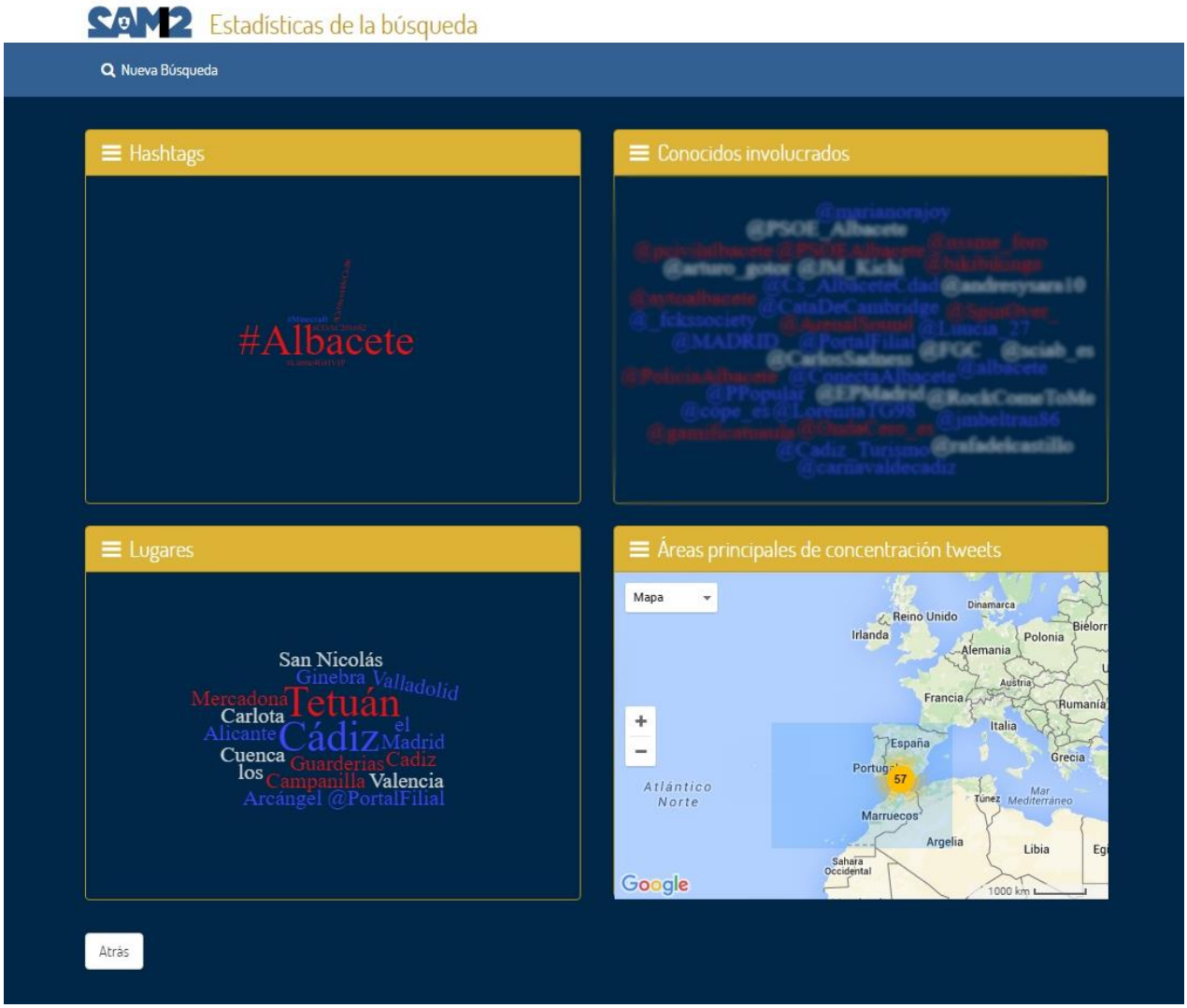

Figura 6-6: Estadísticas en SAMI2

Se puede apreciar en esta captura un resumen sobre la búsqueda realizada. La cuál está formada por tres nubes de etiquetas y un mapa con todas las publicaciones.

El mapa es interactivo, por lo que se puede acercar la vista y al hacerlo aparecerán markers creados específicamente para este proyecto con el logotipo de Twitter si la publicación ha sido publicada utilizando la ubicación del dispositivo o con un interrogante si la ubicación es la que el usuario introdujo a la hora de crear su perfil en la red social. Si se hiciese click en el marker aparecería un recuadro con el texto de la publicación.

En las nubes de etiquetas aparecerán en mayor tamaño los hashtags, menciones o lugares más mencionados en su zona correspondiente.

7 Conclusiones y trabajo futuro

En esta parte del documento se tratarán las conclusiones a las que se han llevado a cabo después de realizar este desarrollo, el cuál es completamente funcional y operativo. Posteriormente se valorará el trabajo futuro que podría añadirse a este sistema para mejorar o incrementar sus posibilidades.

7.1 Conclusiones

El principal objetivo de este desarrollo era proporcionar un sistema capaz de ayudar en la búsqueda de información en las redes sociales, más concretamente enfocado hacia el ámbito policial para la prevención y detección de contenido o actividades ilegales. Con este primordial y fundamental requisito, al finalizar el desarrollo se ha podido comprobar que ésta herramienta puede ser completamente útil y satisfactoria realizando esta labor.

Cabe destacar la rapidez de búsqueda (menor de 15 segundos aproximadamente) y la muestra de resultados por pantalla unido a una interfaz amigable y muy intuitiva, con la que solamente con unas pocas interacciones el usuario ya tiene disponible los datos de la búsqueda solicitada además de diversa información muy útil que puede ampliar según las necesidades de la búsqueda para explorarla en mayor profundidad. El nivel de inmersión en la búsqueda llega a tal nivel que desde la propia web se puede redirigir a la web de Twitter para ver la/s publicación/es de primera mano, con lo que la experiencia de usuario se ve gratamente satisfecha.

Como valor añadido también se ofrece una vista de carácter general mostrando la cantidad de tuits analizados, así como el número de coincidencias que tuvo la búsqueda o la cantidad de menciones a usuarios y hashtags involucrados, con lo que da una visión global de mayor información en menor espacio y tiempo.

Otro punto a destacar es la posibilidad de filtrado dentro de la propia búsqueda realizada. Este filtrado a través de hashtags, menciones a usuarios proporciona poder filtrar las publicaciones resultado más específicamente.

7.2 Trabajo futuro

Como trabajo futuro se propone:

- Avisador mediante correo electrónico cuando alguna de las máquinas se estén quedando sin memoria.

- Mostrar los emoticonos de las publicaciones ya que actualmente se muestran caracteres que no aportan ninguna información.

- Analizador de sentimientos. Mediante el análisis de los emoticonos en la publicación o el sentido o énfasis del texto.

8 Referencias

[1] **HI Iberia**. Empresa donde se desarrolló el proyecto descrito. http://www.hi-iberia.es/

[2] **SAMI2.** Nombre del proyecto descrito en este TFG. sami2-project.eu

[3] **MeteorJs**. Framework Javascript basado en NodeJs. https://www.meteor.com/

[4] **Página de estadísticas de Twitter.** http://www.internetlivestats.com/twitter-statistics/

[5] **Limitar acceso de tuits violentos.** http://www.diarioinformacion.com/vida-y-estilo/tecnologia/2015/04/21/twitter-detectar-tuits-violentos/1623420.html

[6] **Detectar terremotos.** http://thenewstack.io/detecting-earthquakes-twitter-elasticsearch/

[7] **Detectar contenido para proteger a adolescentes.** **https://faculty.ist.psu.edu/xu/papers/Chen_etal_SocialCom_2012.pdf**

[8] **Moderar publicaciones.** http://www.theguardian.com/media/2012/oct/09/dpp-criminal-tweets-facebook-posts

9 Glosario

Algoritmo de hash	Procedimiento informático en el cúal proporciona una salida de tamaño fijo de caracteres sobre una entrada de tamaño indefinido de caracteres
Anonimizar	Hacer que información privada y personal sea anónima
API	Application Programming Interface
Big data	Concepto referenciado al almacenado y procesado de un gran volume de datos
Bounding box	Área definida por dos latitudes y longitudes geográficas
Bugs	Error detectado en un software informático que produce un efecto no deseado
Ciberacoso	Uso de la medios digitales e internet para acosar a una persona
Crawler	Software informático que inspecciona páginas web con el propósito de indexer la información
Framework	Herramienta que sirve de soporte o base para el desarrollo software en un determinado lenguaje. Se suele incluir funcionalidades comunes y básicas
Indexador	Software informático que almacena el contenido mediante índices o palabras clave para una rápida recuperación mediante una búsqueda
Javadoc	Herramienta del lenguaje de programación Java para el desarrollo y visualización de la documentación en ese lenguaje
Log	Registro que produce un software informático en su ejecución. En castellano podría traducirse por bitácora
Markers	Marcadores de ubicación utilizados por google en sus mapas
Naive Bayes	Clasificador de datos basado en la estadística
NoSQL	Base de datos orientada a documentos. Como lenguaje de consulta no utiliza SQL
Nube de tags	Nube de etiquetas
Publicación	Referente en este documento a publicaciones de la red social Twitter
Replica set	Configuración específica del modo de replicación de datos de la base de datos MongoDB
Responsive	En diseño web es el diseño que se adapta al dispositivo que se está usando para la visualización
Scroll	Desplazamiento de contenido en la pantalla de un dispositivo electrónico
Scrum	Una forma de desarrollar software
SessionStorage	Tecnología de los navegadores actuales para el almacenamiento de datos mientras la pestaña del navegador está abierta

Software	Programa informático
Streaming	Distribución de contenido digital con la peculiaridad que el receptor consume el contenido recibido mientras continua recibiendo más.
Tuit	Traducción del término Tweet al que se relaciona con una publicación de la red social Twitter
Tuits	Plural de tuit

10 Anexos

En este último apartado se expondrá información complementaria que no encajaría con los puntos tratados anteriormente.

10.1 Otros posibles usos

Este sistema aunque su diseño ha sido orientado hacia el uso específico de un usuario final puede tener otros posibles usos debido a la naturaleza de su construcción.

- **Filtrado de noticias**: Un uso alternativo sería la búsqueda de información sobre un tema de particular interés para el usuario. Es decir, sería como poder acceder a diferentes tipos de noticias de un periódico (financieras, tecnológicas…) pero con el aliciente de poder disponer de ellas en tiempo real. De esta manera se estaría realmente informados sobre el tema que más nos interese, con la ventaja de poder encaminar o mejorar su búsqueda hacia nuestras preferencias si además se utiliza el mecanismo de evaluación de resultados.

- **Obtener opinión de usuarios**: Durante los últimos años los medios y canales de comunicación optan por utilizar un hashtag identificativo para que los usuarios de la red social puedan manifestar su opinión al instante. Esto puede observarse en capítulos en series de televisión, programas de entretenimiento, marcas de productos comerciales… Y a un sinfín de aplicaciones más.

  Hasta ahora para poder leer las opiniones de los usuarios se tiene que buscar en Twitter el hashtag correspondiente pero con este sistema simplemente con añadir un perfil de búsqueda con palabras clave relacionadas y hashtags, se obtendrá un mayor número de publicaciones de los usuarios dando su opinión. Esto puede ser de gran ayuda sobre todo cuando una marca acaba de lanzar un producto y no sabe la reacción de su clientela o público.

  Utilizando esta herramienta, se obtiene las impresiones del cliente o del usuario para en caso de fallo en el lanzamiento, poderlo corregir lo antes posible o incluir mejoras que los usuarios de la red social publiquen en su cuenta.